A gentle introduction to the discrete Laplace method for estimating Y-STR haplotype frequencies

Mikkel Meyer Andersen
mikl@math.aau.dk
Department of Mathematical Sciences
Aalborg University
Denmark

Poul Svante Eriksen
svante@math.aau.dk
Department of Mathematical Sciences
Aalborg University
Denmark

Niels Morling
niels.morling@sund.ku.dk
Section of Forensic Genetics
Department of Forensic Medicine
Faculty of Health and Medical Sciences
University of Copenhagen
Denmark

Abstract

Y-STR data simulated under a Fisher-Wright model of evolution with a single-step mutation model turns out to be well predicted by a method using discrete Laplace distributions.

Contents

1 Introduction

This tutorial introduces the discrete Laplace method for estimating Y-STR haplotype frequencies as described by Andersen et al. [2013].

To accomplish this, we demonstrate a number of examples using `R` [R Development Core Team, 2012]. The code examples look like the following that loads the `disclap` package [Andersen and Eriksen, 2013a] which is needed for the following examples:

```
library(disclap)
```

If you do not have installed the `disclap` package, please visit `http://cran.r-project.org/package=disclap`.

2 The discrete Laplace distribution

The discrete Laplace distribution is a probability distribution like e.g. the binomial distribution or the normal/Gaussian distribution.

The discrete Laplace distribution has two parameters: a dispersion parameter $0 < p < 1$ and a location parameter $y \in \mathbb{Z} = \{\ldots, -2, -1, 0, 1, 2, \ldots\}$.

Let $X \sim DL(p, y)$ denote that the random variable X follows a discrete Laplace distribution with dispersion parameter $0 < p < 1$ and location parameter y. Then a realisation of the random variable, $X = x$, can be any integer in $\mathbb{Z}$. The random variable X has the probability mass function given by

$$f(X = x; p, y) = \frac{1 - p}{1 + p} \cdot p^{|x-y|} \quad \text{for } x \in \mathbb{Z}.$$

As seen, only the absolute value of $x - y$ is used. This means that the probability mass function is symmetric around y.

Let us try to plot the probability mass function $f(X = x; p, y)$ for $p = 0.3$ and $y = 13$ from $x = 8$ to $x = 18$:

```
p <- 0.3
y <- 13
x <- seq(8, 18, by = 1)
barplot(ddisclap(x - y, p), names = x, xlab = "x, e.g. Y-STR allele",
    ylab = paste("Probability mass, f(X = x; ", p, ", ", y, ")", sep = ""))
```

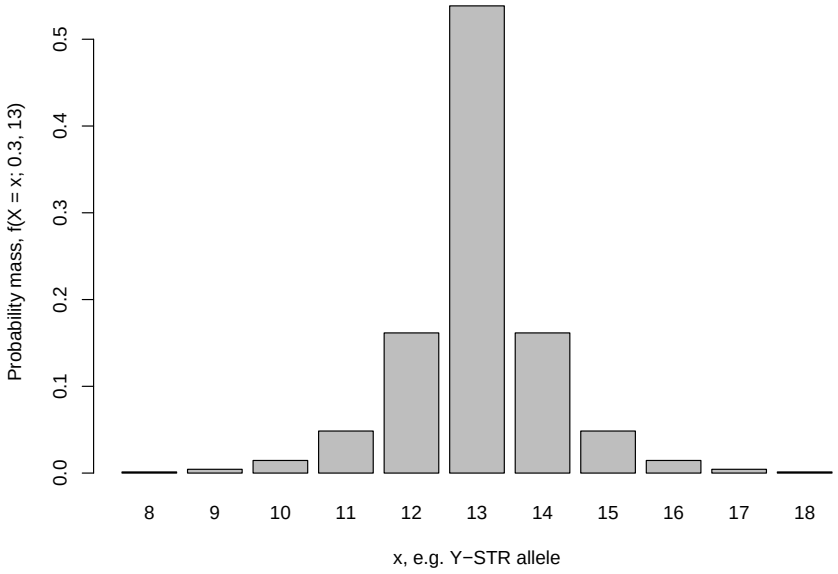

Figure 1: The probability mass function, $f(X = x; p, y)$, for the discrete Laplace distribution with dispersion parameter $p = 0.3$ and location parameter $y = 13$ from $x = 8$ to $x = 18$.

We plot the distribution for values of x from 8 to 18 as there is almost no probability mass outside these values. We can find out how much of the probability mass that we have plotted:

```
sum(ddisclap(x - y, p))
```

```
## [1] 0.9989
```

Thus, only 0.0011 of the probability mass is outside $\{8, 9, \ldots, 17, 18\}$.

If we have a sample of realisations from $X \sim DL(p, y)$ denoted by $\{x_i\}_{i=1}^n$, then maximum likelihood estimates are given by the following quantities [Andersen et al., 2013]:

$$\begin{aligned}
\hat{y} &= \text{median}\{x_i\}_{i=1}^n, \\
\hat{\mu} &= \frac{1}{n}\sum_{i=1}^{n} |x_i - \hat{y}| \text{ and} \\
\hat{p} &= \hat{\mu}^{-1}\left(\sqrt{\hat{\mu}^2 + 1} - 1\right).
\end{aligned}$$

Example:

```r
set.seed(1)  # Makes it possible to reproduce the simulation results
p <- 0.3  # Dispersion parameter
y <- 13  # Location parameter
x <- rdisclap(100, p) + y  # Generate a sample using the rdisclap function

y.hat <- median(x)
y.hat
```

```
## [1] 13
```

```r
mu.hat <- mean(abs(x - y.hat))
mu.hat
```

```
## [1] 0.57
```

```r
p.hat <- mu.hat^(-1) * (sqrt(mu.hat^2 + 1) - 1)
p.hat  # We expect 0.3
```

```
## [1] 0.265
```

```r
# The observed distribution of d's
tab <- prop.table(table(x))
tab
```

```
## x
##   10   11   12   13   14   15   16
## 0.01 0.03 0.15 0.55 0.20 0.05 0.01
```

This can be plotted against the expected counts as follows:

```
plot(1:length(tab), ddisclap(as.integer(names(tab)) - y.hat, p.hat),
  type = "h", col = "#999999", lend = "butt", lwd = 50,
  xlab = "x, e.g. Y-STR allele", ylab = "Probability mass", axes = FALSE)
axis(1, at = 1:length(tab), labels = names(tab))
axis(2)
points(1:length(tab), tab, type = "h", col = "#000000",
  lend = "butt", lwd = 25)
legend("topright", c("Estimated distribution", "Observations"),
  pch = 15, col = c("#999999", "#000000"))
```

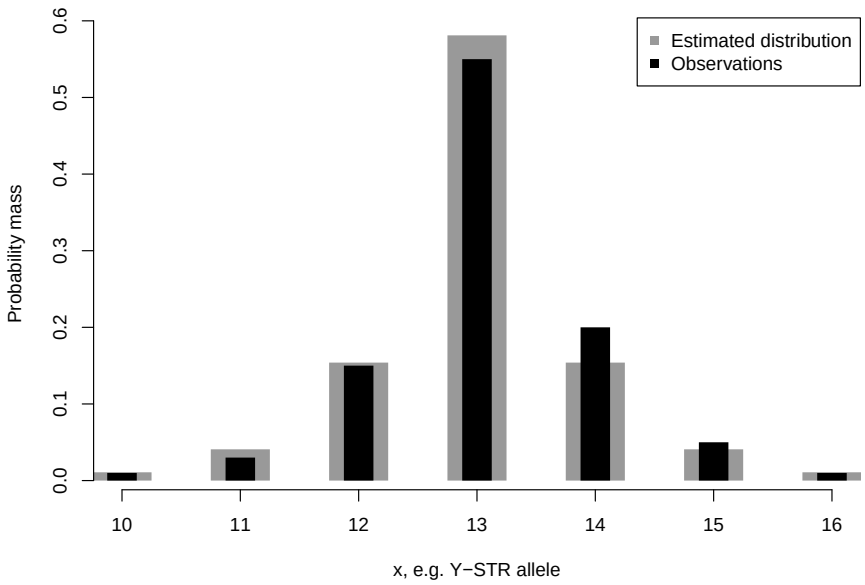

Figure 2: Observed frequencies of the x's compared to a discrete Laplace distribution with parameters estimated from the sample.

3 Mixtures of multivariate, marginally independent, discrete Laplace distributions

Assume a very simple 'haplotype' with only one locus. Also assume a simple and isolated population. Then, it is reasonable to assume that there is a modal/central Y-STR allele, y, and that all the alleles are distributed around this allele.

If we go back to Figure 2, this can be illustrated by $y = 13$ as the central Y-STR allele and a distribution around $y = 13$ with shorter and longer alleles.

To begin with, it might seem a bit overwhelming that Y-STR alleles should follow a simple probabiity distribution such as the discrete Laplace distribution. But surprisingly, it is actually a good approximation as demonstrated by Andersen et al. [2013].

We have haplotypes with several loci. When we assess multiple loci haplotypes, we assume that mutations happen independently across loci. Each locus has its own discrete Laplace distribution of allele probabilities, and the probability of a haplotype is the product of probabilities across loci. This gives a multivariate discrete Laplace distribution, where the marginals (that is, at each locus) are independent, discrete Laplace distributions.

Just as before, for a one locus haplotype, we can assume that there is a modal/central Y-STR profile with r loci, $y = (y_1, y_2, \ldots, y_r)$, and all the alleles are distributed around this profile. We also assume that the discrete Laplace distribution at each locus has its own parameter, where p_k is the parameter at the k^{th} locus. Normally, the central Y-STR profile, y, would also be regarded as parameters.

As before, let $f(x; p, y)$ be the probability mass function of a discrete Laplace distribution. We define an observation $X = (X_1, X_2, \ldots, X_r)$ to be from a multivariate distribution of independent, discrete Laplace distributions when the probability of observing $X = x$ is

$$\prod_{k=1}^{r} f\left(x_k; p_k, y_k\right). \tag{1}$$

This corresponds to that the individual X has mutated away from y independently at each locus.

Now, we have one more generalisation. A population may have several subpopulations, e.g. introduced by migration or by evolution. This means that we need to have a mixture of multivariate distributions with marginally independent, discrete Laplace distributions. Each component in the mixture represents a subpopulation. We define an observation $X = (X_1, X_2, \ldots, X_r)$ to be from a mixture of multivariate, marginally independent, discrete Laplace distributions, when the probability of observing $X = x$ is

$$\sum_{j=1}^{c} \tau_j \prod_{k=1}^{r} f\left(x_k; p_{jk}, y_{jk}\right), \tag{2}$$

where τ_j is the a priori probability for originating from the j'th subpopulation. Thus, the parameters of this mixture model are $\{y_j\}_{j=1}^{c}$ with $y_j = (y_{j1}, y_{j2}, \ldots, y_{jr})$ as the central haplotype of the j^{th} subpopulation, $\{\tau_j\}_{j=1}^{c}$ and $\{p_{jk}\}_{\substack{j\in\{1,2,\ldots,c\}\\ k\in\{1,2,\ldots,r\}}}$ (the parameters for each discrete Laplace distribution).

We assume that p_{jk} depends on locus and subpopulation, such that $\log p_{jk} = \omega_j + \lambda_k$. This means that there is an additive effect of locus, λ_k, and an additive effect of subpopulation, ω_j.

More theory on finite mixture distributions is given by Titterington et al. [1987].

3.1 Haplotype frequency prediction

When we have estimated the parameters of a mixture of multivariate, marginally independent, discrete Laplace distributions (this will be shown in the next section), we can use these to estimate haplotype frequencies.

Given estimates of subpopulation centers $\{\hat{y}_j\}_j$, dispersion parameters $\{\hat{p}_{jk}\}_{j,k}$ and prior probabilities $\{\hat{\tau}_j\}_j$, the haplotype frequency of a haplotype $x = (x_1, x_2, \ldots, x_r)$ with $x_k \in \mathbb{Z}$ for $k \in \{1, 2, \ldots, r\}$ can be estimated as

$$\hat{p}(x) = \sum_{j=1}^{c} \hat{\tau}_j \prod_{k=1}^{r} f\left(x_k; \hat{p}_{jk}, \hat{y}_{jk}\right). \tag{3}$$

Thus, we simply use the estimated parameters in Equation (2) to obtain Equation 3.

4 Estimating parameters

In this section we demonstrate how to estimate the parameters in a mixture of multivariate, independent, discrete Laplace distributions. This can for example be used to estimate Y-STR haplotype frequencies.

First, the `R` package `disclapmix` [Andersen and Eriksen, 2013b, Andersen et al., 2013] for analysing a mixture of multivariate, independent, discrete Laplace distributions must be loaded:

```
library(disclapmix)
```

If you do not have the `disclapmix` package installed, please visit `http://cran.r-project.org/package=disclapmix`.

This package supplies the function `disclapmix` for estimating the parameters in a mixture of multivariate, marginally independent, discrete Laplace distributions with probability mass function given in Equation (2). We will refer to this as 'the discrete Laplace method'.

4.1 Data from marginally independent, discrete Laplace distributions

Now, we revisit the example leading to Figure 2 and add two more loci with different dispersion and location parameters. We then analyse the randomly generated values from independent, discrete Laplace distributions with a probability mass function as given in Equation (1).

```
set.seed(1)
n <- 100  # number of individuals

# Locus 1
p1 <- 0.3  # Dispersion parameter
m1 <- 13  # Location parameter
d1 <- rdisclap(n, p1) + m1  # Generate a sampling using the rdisclap function

# Locus 2
p2 <- 0.4
m2 <- 14
d2 <- rdisclap(n, p2) + m2
```

```r
# Locus 3
p3 <- 0.5
m3 <- 15
d3 <- rdisclap(n, p3) + m3

db <- cbind(d1, d2, d3)
head(db)
```

```
##      d1 d2 d3
## [1,] 14 15 16
## [2,] 12 12 17
## [3,] 13 13 15
## [4,] 13 13 15
## [5,] 14 12 15
## [6,] 13 15 15
```

```r
fit <- disclapmix(db, centers = 1, verbose = 0)
```

We can then look at the estimated location parameters, $y = (y_1, y_2, y_3)$:

```r
fit$best.fit$disclapdata$y
```

```
##      [,1] [,2] [,3]
## [1,]   13   14   15
```

And the estimated dispersion parameters, (p_1, p_2, p_3):

```r
fit$best.fit$pred.ps
```

```
##      1      2      3
## 0.2650 0.4369 0.5167
```

As seen, the estimated dispersion location parameters are well estimated. The dispersion parameters are also quite close to the ones used to generate the data.

4.2 Data from a Fisher-Wright population

Andersen et al. [2013] simulated populations following the Fisher-Wright model of evolution [Fisher, 1922, 1930, 1958, Wright, 1931, Ewens, 2004] with assumptions of primarily neutral, single-step mutations of STRs [Ohta and Kimura, 1973]. From these populations, data sets were sampled. Using the discrete Laplace method for estimating haplotype frequencies, the method worked rather well.

This is worth highlighting: Data was simulated under a completely different model than that used for inference afterwards. The data was simulated under a population model (Fisher-Wright model of evolution) with a certain mutation model (single-step mutation model). Inference was made assuming that the data was from a mixture of multivariate, marginally independent, discrete Laplace distributions.

One of the reasons that the discrete Laplace distribution predicts data from a Fisher-Wright model of evolution with a single-step mutation model is due to the fact that it approximates certain properties of this population and mutation model [Caliebe et al., 2010]. This is also explained by Andersen et al. [2013].

Now, let us try simulating a Fisher-Wright population and analyse it with the discrete Laplace method. To simulate the population, the `R` package **`fwsim`** [Andersen and Eriksen, 2012a,b] is loaded:

```
library(fwsim)
```

If you do not have the **`fwsim`** package installed, please visit **`http://cran.r-project.org/package=fwsim`**.

We then simulate a population consisting of Y-STR profiles:

```
set.seed(1)
generations <- 100
population.size <- 1e+05
number.of.loci <- 7
mutation.rates <- seq(0.001, 0.01, length.out = number.of.loci)
mutation.rates
```

```
## [1] 0.0010 0.0025 0.0040 0.0055 0.0070 0.0085 0.0100
```

```
sim <- fwsim(g = generations, k = population.size, r = number.of.loci,
    mu = mutation.rates, trace = FALSE)
pop <- sim$haplotypes
```

Note, that the mutation rates are different for each locus (ranging from 0.001 to 0.01). The location parameter is 0 for all loci by default. This can be changed afterwards without loosing or adding any information. Below, we change it to be $y = (14, 12, 28, 22, 10, 11, 13)$:

```
y <- c(14, 12, 28, 22, 10, 11, 13)
for (i in 1:number.of.loci) {
    pop[, i] <- pop[, i] + y[i]
}
head(pop)
```

```
##   Locus1 Locus2 Locus3 Locus4 Locus5 Locus6 Locus7 N
## 1     12     12     28     22     10     11     13 3
## 2     14     11     26     20      9     11     13 1
## 3     13     11     26     22     10     10     13 4
## 4     14     11     26     22      8     10     13 2
## 5     14     11     26     22      9     10     12 2
## 6     14     11     26     23     10     10     11 2
```

Then, y is the most frequent 10 locus Y-STR haplotype in Denmark according to **`http://www.yhrd.org`** (on March 26, 2013) restricted to the 7 loci minimal haplotype.

The column `N` is the number of individuals in the population with that Y-STR haplotype. Summing column `N` reveals that there is not exactly **`population.size`** individuals due to that

the population size is stochastic (refer to Andersen and Eriksen [2012b] for the details).

We can then calculate the population frequency for each haplotype:

```
pop$PopFreq <- pop$N/sum(pop$N)
```

Let us draw a data set where each haplotype is drawn relatively to its population frequency:

```
set.seed(1)
n <- 500  # Data set size
types <- sample(x = 1:nrow(pop), size = n, replace = TRUE, prob = pop$N)
types.table <- table(types)

alpha <- sum(types.table == 1)
alpha/n  # Singleton proportion
```

```
## [1] 0.492
```

```
dataset <- pop[as.integer(names(types.table)), ]
dataset$Ndb <- types.table
head(dataset)
```

```
##     Locus1 Locus2 Locus3 Locus4 Locus5 Locus6 Locus7   N   PopFreq Ndb
## 9       14     11     26     23     10      8     12   2 1.924e-05   1
## 103     14     11     28     19      9     10     12   1 9.619e-06   1
## 146     14     11     28     21     10     11     13 187 1.799e-03   3
## 229     14     11     27     21     11     12     12   6 5.771e-05   1
## 271     14     11     28     22      7     11     12  14 1.347e-04   1
## 273     14     11     28     22      8     11     12   6 5.771e-05   1
```

```
db <- pop[types, 1:number.of.loci]
head(db)
```

```
##      Locus1 Locus2 Locus3 Locus4 Locus5 Locus6 Locus7
## 1162     13     12     30     22      8     11     11
## 3053     14     12     28     22     10     11     14
## 2773     14     13     28     21     10     10     14
## 1544     14     12     28     22      9     11     14
## 3239     14     12     28     22     11     11     14
## 1120     14     12     28     22      9     10     14
```

Then, analyse it:

```
fit <- disclapmix(db, centers = 1, verbose = 0)

# Estimated location parameters
fit$best.fit$disclapdata$y
```

```
##      [,1] [,2] [,3] [,4] [,5] [,6] [,7]
```

```
## [1,]   14   12   28   22   10   11   13


# Estimated dispersion parameters
fit$best.fit$pred.ps

##      1      2      3      4      5      6      7
## 0.0469 0.1260 0.1589 0.1827 0.2453 0.2817 0.3160
```

Let us compare the mutation rates with the dispersion parameters in the discrete Laplace distributions:

```
plot(mutation.rates, fit$best.fit$pred.ps, xlab = "Mutation rate",
    ylab = "Estimated dispersion parameter")
```

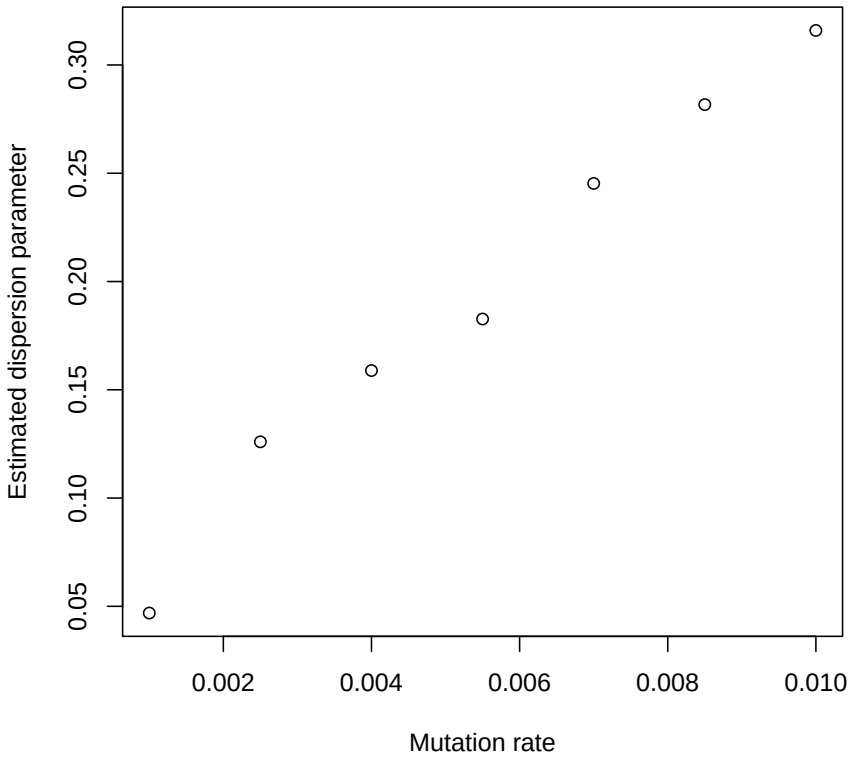

Figure 3: The relationship between the mutation rate in a Fisher-Wright population and the estimated dispersion parameters using the discrete Laplace method.

As expected, there is a connection between the mutation rate and the dispersion parameter (the exact connection is not known).

It is possible to predict a population frequency with the `predict` function as shown in Equation (3). This can be used to see how well the population frequency is predicted for each

unique haplotype in the dataset (obtained by using `dataset` instead of `db`):

```
pred.popfreqs <- predict(fit$best.fit, newdata = dataset[, 1:number.of.loci])
plot(dataset$PopFreq, pred.popfreqs, log = "xy",
  xlab = "True population frequency",
  ylab = "Estimated population frequency")
abline(a = 0, b = 1, lty = 1)
legend("bottomright", "y = x (predicted = true)", lty = 1)
```

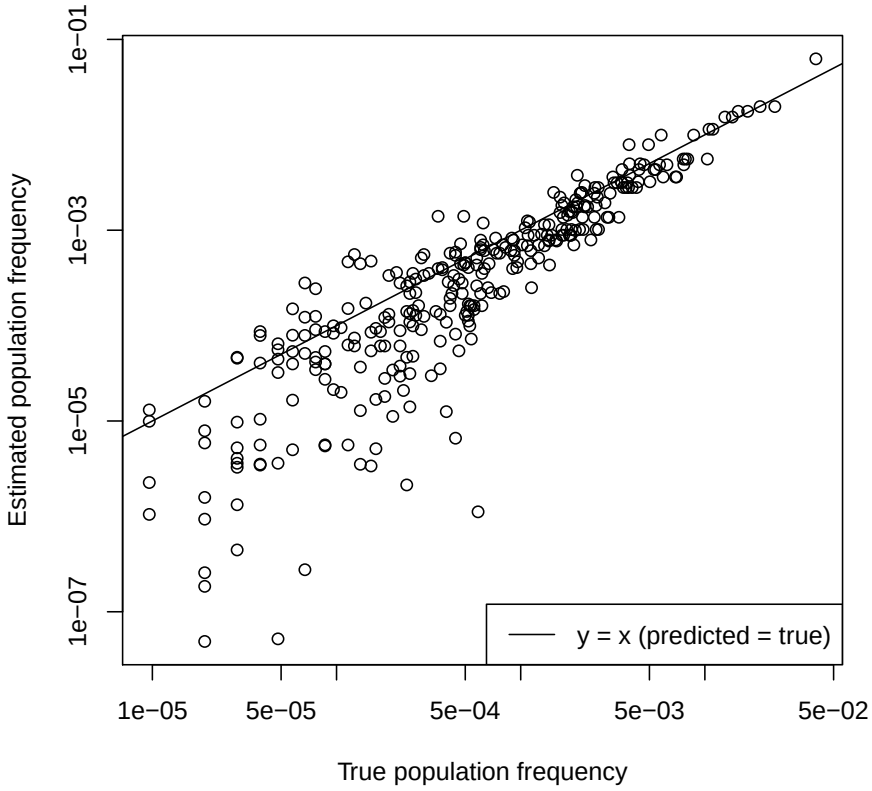

Figure 4: The relationship between the true population frequency and the predicted population frequency using the discrete Laplace method.

4.3 Data from a mixture of two Fisher-Wright populations

Here, we show how to analyse a dataset from a mixture of two populations. First, we simulate two populations (note the different mutation rates and location parameters, where the location parameters again are changed afterwards without loosing or adding any information):

```
set.seed(1)

# Common parameters
```

```r
generations <- 100
population.size <- 1e+05
number.of.loci <- 7

mu1 <- seq(0.001, 0.005, length.out = number.of.loci)
sim1 <- fwsim(g = generations, k = population.size, r = number.of.loci,
    mu = mu1, trace = FALSE)
pop1 <- sim1$haplotypes
y1 <- c(14, 12, 28, 22, 10, 11, 13)
for (i in 1:number.of.loci) pop1[, i] <- pop1[, i] + y1[i]

mu2 <- seq(0.005, 0.01, length.out = number.of.loci)
sim2 <- fwsim(g = generations, k = population.size, r = number.of.loci,
    mu = mu2, trace = FALSE)
pop2 <- sim2$haplotypes
y2 <- c(14, 13, 29, 23, 11, 13, 13)
for (i in 1:number.of.loci) pop2[, i] <- pop2[, i] + y2[i]
```

Here, just as $y_1 = (14, 12, 28, 22, 10, 11, 13)$ are the alleles from most frequent haplotype, then $y_2 = (14, 13, 29, 23, 11, 13, 13)$ are the alleles from the second most frequent haplotype.

Then we sample a data set with an expected proportion of 20% from the first population and 80% from the second population:

```r
set.seed(1)
n <- 500  # Data set size

n1 <- rbinom(1, n, 0.2)
c(n1, n1/n)
```

```
## [1] 102.000   0.204
```

```r
n2 <- n - n1
c(n2, n2/n)
```

```
## [1] 398.000   0.796
```

```r
types1 <- sample(x = 1:nrow(pop1), size = n1, replace = TRUE, prob = pop1$N)
db1 <- pop1[types1, 1:number.of.loci]

types2 <- sample(x = 1:nrow(pop2), size = n2, replace = TRUE, prob = pop2$N)
db2 <- pop2[types2, 1:number.of.loci]

db <- rbind(db1, db2)

# Singleton proportion
sum(table(apply(db, 1, paste, collapse = ";")) == 1)/n
```

```
## [1] 0.672
```

Now, we analyse the data set trying 1 to 5 subpopulations. Afterwards, we analyse the optimal number of subpopulations using the BIC (Bayesian Information Criteria) by Schwarz [1978]:

```
fit <- disclapmix(db, centers = 1:5, use.parallel = TRUE, verbose = 0)
```

The BIC values are:

```
sapply(fit$fits, extractMarginalBIC)

## [1] 9487 8600 8646 8700 8748
```

Here, the optimal number of subpopulations is 2. The estimated parameters for this optimal number of subpopulations are available at the `best.fit`-slot:

```
fit$best.fit

## disclapmixfit from 500 observations on 7 loci with 2 centers.


# Estimated a priori probability of originating from each
# subpopulation
fit$best.fit$disclapdata$tau

## [1] 0.2126 0.7874


# Estimated location parameters
fit$best.fit$disclapdata$y

##         Locus1 Locus2 Locus3 Locus4 Locus5 Locus6 Locus7
## 1577.24     14     12     28     22     10     11     13
## 8158.2      14     13     29     23     11     13     13


# Estimated dispersion parameters for each subpopulation
fit$best.fit$pred.ps

## [[1]]
##      1      2      3      4      5      6      7
## 0.1029 0.1083 0.1213 0.1353 0.1458 0.1587 0.1595
##
## [[2]]
##      1      2      3      4      5      6      7
## 0.1896 0.1997 0.2234 0.2494 0.2686 0.2924 0.2938
```

The estimated location parameters are the same as those used for generating the data. Also, the values of τ_j, the a priori probability of originating from the j^{th} subpopulation, are consistent with the mixture proportions of 0.204 and 0.796.

We can also calculate the predicted population frequencies (using the mixture proportions

0.204 and 0.796):

```
pop1$PopFreq <- pop1$N/sum(pop1$N)
pop2$PopFreq <- pop2$N/sum(pop2$N)

types1.table <- table(types1)
types2.table <- table(types2)

dataset1 <- pop1[as.integer(names(types1.table)), ]
dataset1$Ndb <- types1.table
sum(dataset1$Ndb)
```

```
## [1] 102
```

```
dataset2 <- pop2[as.integer(names(types2.table)), ]
dataset2$Ndb <- types2.table
sum(dataset2$Ndb)
```

```
## [1] 398
```

```
dataset <- merge(x = dataset1, y = dataset2, by = colnames(db), all = TRUE)
dataset[is.na(dataset)] <- 0

dataset$MixPopFreq <- (n1/n) * dataset$PopFreq.x + (n2/n) * dataset$PopFreq.y

dataset$Type <- "Only from pop1"
dataset$Type[dataset$Ndb.y > 0] <- "Only from pop2"
dataset$Type[dataset$Ndb.x > 0 & dataset$Ndb.y > 0] <- "Occurred in both"
dataset$Type <- factor(dataset$Type)
```

We can now compare the predicted frequencies with the population frequency:

```
pred.popfreqs <- predict(fit$best.fit, newdata = dataset[, 1:number.of.loci])
plot(dataset$MixPopFreq, pred.popfreqs, log = "xy", col = dataset$Type,
  xlab = "True population frequency",
  ylab = "Estimated population frequency")
abline(a = 0, b = 1, lty = 1)
legend("bottomright", c("y = x (predicted = true)", levels(dataset$Type)),
  lty = c(1, rep(-1, 3)), col = c("black", 1:length(levels(dataset$Type))),
  pch = c(-1, rep(1, 3)))
```

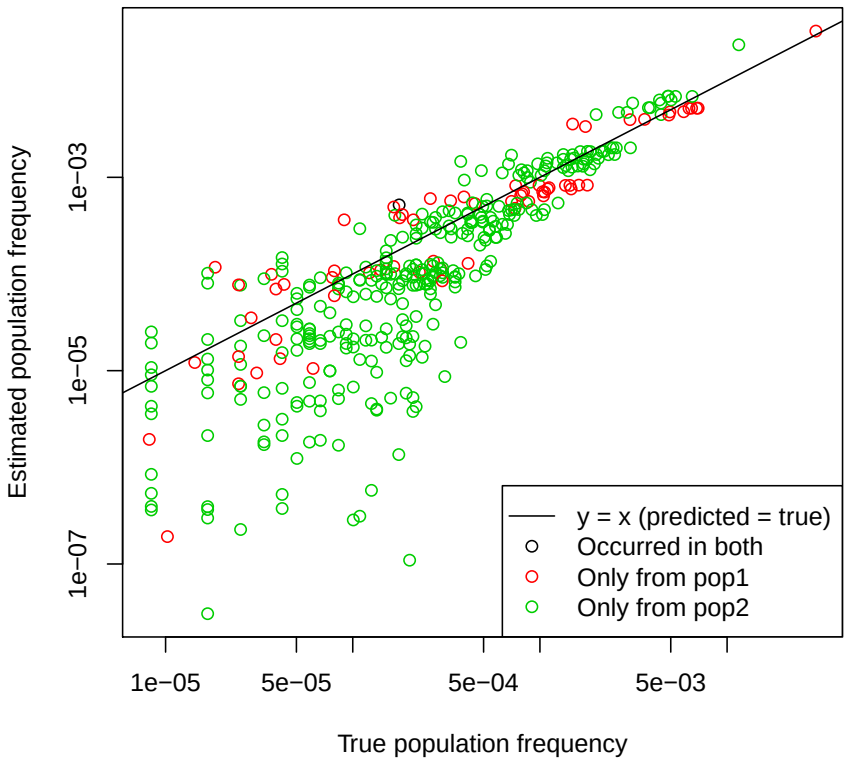

Figure 5: The relationship between the true population frequency and the predicted population frequency using the discrete Laplace method.

5 Concluding remarks

We have shown how to analyse Y-STR population data using the discrete Laplace method described by Andersen et al. [2013]. This was done using the freely available and open-source `R` packages `disclap`, `fwsim` and `disclapmix` that are supported on Linux, MacOS and MS Windows.

One key point made is worth repeating: Data simulated under a population model (e.g.

the Fisher-Wright model of evolution) with a certain mutation model (e.g. the single-step mutation model) can be successfully analysed using the discrete Laplace method making inference assuming that the data is from a mixture of multivariate, independent, discrete Laplace distributions.

References

Mikkel Meyer Andersen and Poul Svante Eriksen. *fwsim: Fisher-Wright Population Simulation*, 2012a. URL `http://CRAN.R-project.org/package=fwsim`. R package version 0.2-5.

Mikkel Meyer Andersen and Poul Svante Eriksen. Efficient forward simulation of fisher-wright populations with stochastic population size and neutral single step mutations in haplotypes. *Preprint*, 2012b. arXiv:1210.1773.

Mikkel Meyer Andersen and Poul Svante Eriksen. *disclap: Discrete Laplace Family*, 2013a. URL `http://CRAN.R-project.org/package=disclap`. R package version 1.2.

Mikkel Meyer Andersen and Poul Svante Eriksen. *disclapmix: Discrete Laplace mixture inference using the EM algorithm*, 2013b. URL `http://CRAN.R-project.org/package=disclapmix`. R package version 0.3.

Mikkel Meyer Andersen, Poul Svante Eriksen, and Niels Morling. The discrete Laplace exponential family and estimation of Y-STR haplotype frequencies. *Journal of Theoretical Biology*, 2013. In press: `http://dx.doi.org/10.1016/j.jtbi.2013.03.009`.

Amke Caliebe, Arne Jochens, Michael Krawczak, and Uwe Rösler. A Markov Chain Description of the Stepwise Mutation Model: Local and Global Behaviour of the Allele Process. *Journal of Theoretical Biology*, 266(2):336–342, 2010. ISSN 0022-5193.

Warren J. Ewens. *Mathematical Population Genetics*. Springer-Verlag, 2004.

R. A. Fisher. On the Dominance Ratio. *Proc. Roy. Soc. Edin.*, 42:321–341, 1922.

R. A. Fisher. *The Genetical Theory of Natural Selection*. Oxford: Clarendon Press, 1930.

R. A. Fisher. *The Genetical Theory of Natural Selection*. New York: Dover, 2nd revised edition, 1958.

T. Ohta and M. Kimura. A Model of Mutation Appropriate to Estimate the Number of Electrophoretically Detectable Alleles in a Finite Population. *Genet. Res.*, 22:201–204, 1973.

R Development Core Team. *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria, 2012. URL `http://www.R-project.org`. ISBN 3-900051-07-0.

Gideon Schwarz. Estimating the Dimension of a Model. *Annals of Statistics*, 6(2):461–464, 1978.

D. M. Titterington, A. F. M. Smith, and U. E. Makov. *Statistical Analysis of Finite Mixture Distributions*. Wiley, 1987.

S. Wright. Evolution in Mendelian populations. *Genetics*, 16:97–159, 1931.